

I claim:

1 1. In a computing environment having a connection to a network, computer readable code
2 readable by a computer system in said environment, for enhancing performance of a multithreaded
3 application, comprising:

4 a plurality of client requests for connections;
5 a plurality of worker threads;
6 a subprocess for receiving said plurality of client requests; and
7 a subprocess for implementing a scheduling heuristic to alleviate over-scheduling of said
8 worker threads.

2 2. Computer readable code for enhancing performance of a multithreaded application

according to Claim 1, wherein:

3 a first group of said worker threads are active threads, said first group being comprised of
4 changeable ones of said plurality of worker threads, and having a changeable number of said
5 changeable ones, said changeable number being at least one; and

6 said subprocess for implementing a scheduling heuristic further comprises a subprocess for
7 balancing said changeable number in said first group against a current workload comprised of one
8 or more of said plurality of client requests.

1 3. Computer readable code for enhancing performance of a multithreaded application
2 according to Claim 2, wherein said subprocess for balancing further comprises using an average
3 delay.

1 4. Computer readable code for enhancing performance of a multithreaded application
2 according to Claim 3, wherein said subprocess for balancing further comprises using a maximum
3 delay.

1 5. Computer readable code for enhancing performance of a multithreaded application
2 according to Claim 4, wherein said average delay and said maximum delay are configuration
3 parameters.

6. Computer readable code for enhancing performance of a multithreaded application
according to Claim 2, wherein:

 a second group of said worker threads are blocked threads, said second group being
comprised of ones of said plurality of worker threads which are not in said first group; and
 said blocked threads are stored in a Last-In, First-Out queue.

7. In a computing environment having a connection to a network, computer readable code
readable by a computer system in said environment, for enhancing performance of a multithreaded

application, comprising:

 a subprocess for moving connections from a pending connections queue to a first queue
when each of said connections are accepted;

 a subprocess for moving each of said connections from said first queue to a second queue
when an initial data packet arrives for said connection; and

8 a subprocess for assigning a worker thread to each of said connections on said second
9 queue.

1 8. In a computing environment having a connection to a network, computer readable code
2 readable by a computer system in said environment, for enhancing performance of a multithreaded
3 application, comprising:

4 a subprocess for receiving input from multiple sources; and
5 a subprocess for merging said received input onto a single queue for scheduling. 1

10 9. Computer readable code for enhancing performance of a multithreaded application
11 according to Claim 8, further comprising:

12 a subprocess for moving connections from a pending connections queue to a first queue
13 when each of said connections are accepted;

14 a subprocess for moving each of said connections from said first queue to said single
15 queue when an initial data packet arrives for said connection; and

16 a subprocess for assigning a worker thread to each of said connections on said single
17 queue.

1 10. Computer readable code for enhancing performance of a multithreaded application
2 according to Claim 9, wherein said subprocess for scheduling further comprises:
3 a group of active worker threads comprised of changeable ones of a plurality of worker
4 threads, and having a changeable number of said changeable ones, said changeable number being

5 at least one; and
6 a subprocess for implementing a scheduling heuristic for balancing said changeable number
7 in said active group against a current workload comprised of said client requests stored on said
8 single queue.

1 11. In a computing environment having a connection to a network, computer readable code
2 readable by a computer system in said environment, for enhancing performance of a multithreaded
3 application, comprising:

4 a plurality of persistent connections;

5 a plurality of worker threads;

6 a subprocess for binding selected ones of said persistent connections to selected ones of
7 said worker threads, wherein an execution of said subprocess for binding results in a bound
8 connection; and

9 a subprocess for unbinding selected ones of said bound connections, wherein an execution
10 of said subprocess for unbinding results in an unbound worker thread.

1 12. Computer readable code for enhancing performance of a multithreaded application
2 according to Claim 11, wherein:

3 said subprocess for binding further comprises using a 2-stage queue; and

4 said subprocess for unbinding further comprises using said 2-stage queue.

1 13. Computer readable code for enhancing performance of a multithreaded application

2 according to Claim 12, wherein:

3 . . . said subprocess for binding using said 2-stage queue further comprises:

4 a subprocess for moving each of said persistent connections to said first stage

5 when an initial data packet arrives for said connection;

6 a subprocess for moving each of said persistent connections from said second

7 stage to said first stage when data is received for said connection; and

8 a subprocess for scheduling said persistent connections from said first stage; and

9 said subprocess for unbinding using said 2-stage queue further comprises:

10 a subprocess for moving selected ones of said bound connections from said first

11 stage to said second stage when said selected bound connection goes idle;

12 a subprocess for closing selected ones of said persistent connections in said second

13 stage, responsive to a maximum idle period being exceeded; and

14 a subprocess for making said unbound worker thread available to said subprocess

15 for binding.

14. Computer readable code for enhancing performance of a multithreaded application

2 according to Claim 13, wherein said subprocess for unbinding further comprises:

3 a subprocess for closing further selected ones of said persistent connections in said second

4 stage, responsive to exceeding a maximum number of idle connections.

1 15. A system for enhancing performance of a multithreaded application in a computing

2 environment having a connection to a network, comprising:

3 a plurality of client requests for connections;
4 a plurality of worker threads;
5 means for receiving said plurality of client requests; and
6 means for implementing a scheduling heuristic to alleviate over-scheduling of said worker
7 threads.

1 16. The system for enhancing performance of a multithreaded application according to Claim
2 15, wherein:

3 a first group of said worker threads are active threads, said first group being comprised of
4 changeable ones of said plurality of worker threads, and having a changeable number of said
5 changeable ones, said changeable number being at least one; and

6 said means for implementing a scheduling heuristic further comprises means for balancing
7 said changeable number in said first group against a current workload comprised of one or more
8 of said plurality of client requests.

1 17. The system for enhancing performance of a multithreaded application according to Claim
2 16, wherein said means for balancing further comprises using an average delay.

1 18. The system for enhancing performance of a multithreaded application according to Claim
2 17, wherein said means for balancing further comprises using a maximum delay.

1 19. The system for enhancing performance of a multithreaded application according to Claim

2 18, wherein said average delay and said maximum delay are configuration parameters.

1 20. The system for enhancing performance of a multithreaded application according to Claim

2 16, wherein:

3 a second group of said worker threads are blocked threads, said second group being

4 comprised of ones of said plurality of worker threads which are not in said first group; and

5 said blocked threads are stored in a Last-In, First-Out queue.

1 21. A system for enhancing performance of a multithreaded application in a computing

2 environment having a connection to a network, comprising:

3 means for moving connections from a pending connections queue to a first queue when

4 each of said connections are accepted;

5 means for moving each of said connections from said first queue to a second queue when

6 an initial data packet arrives for said connection; and

7 means for assigning a worker thread to each of said connections on said second queue.

1 22. A system for enhancing performance of a multithreaded application in a computing

2 environment having a connection to a network, comprising:

3 means for receiving input from multiple sources; and

4 means for merging said received input onto a single queue for scheduling.

1 23. The system for enhancing performance of a multithreaded application according to Claim

2 22, further comprising:

3 means for moving connections from a pending connections queue to a first queue when

4 each of said connections are accepted;

5 means for moving each of said connections from said first queue to said single queue when

6 an initial data packet arrives for said connection; and

7 means for assigning a worker thread to each of said connections on said single queue.

1 24. The system for enhancing performance of a multithreaded application according to Claim

2 23, wherein said means for scheduling further comprises:

3 a group of active worker threads comprised of changeable ones of a plurality of worker
4 threads, and having a changeable number of said changeable ones, said changeable number being
5 at least one; and

6 means for implementing a scheduling heuristic for balancing said changeable number in
7 said active group against a current workload comprised of said client requests stored on said
8 single queue.

1 25. A system for enhancing performance of a multithreaded application in a computing
2 environment having a connection to a network, comprising:

3 a plurality of persistent connections;

4 a plurality of worker threads;

5 means for binding selected ones of said persistent connections to selected ones of said

6 worker threads, wherein an execution of said subprocess for binding results in a bound

7 connection; and

8 means for unbinding selected ones of said bound connections, wherein an execution of
9 said subprocess for unbinding results in an unbound worker thread.

1 26. The system for enhancing performance of a multithreaded application according to Claim
2 25, wherein:

3 said means for binding further comprises using a 2-stage queue; and
4 said means for unbinding further comprises using said 2-stage queue.

5 27. The system for enhancing performance of a multithreaded application according to Claim
6 26, wherein:

7 said means for binding using said 2-stage queue further comprises:
8 means for moving each of said persistent connections to said first stage when an
9 initial data packet arrives for said connection;

10 means for moving each of said persistent connections from said second stage to
11 said first stage when data is received for said connection; and

12 means for scheduling said persistent connections from said first stage; and
13 said means for unbinding using said 2-stage queue further comprises:

14 means for moving selected ones of said bound connections from said first stage to
15 said second stage when said selected bound connection goes idle;

16 means for closing selected ones of said persistent connections in said second stage,
17 responsive to a maximum idle period being exceeded; and

14 means for making said unbound worker thread available to said subprocess for
15 binding.

1 28. The system for enhancing performance of a multithreaded application according to Claim
2 27, wherein said means for unbinding further comprises:

3 means for closing further selected ones of said persistent connections in said second stage,
4 responsive to exceeding a maximum number of idle connections.

1 29. A method for enhancing performance of a multithreaded application in a computing
2 environment having a connection to a network, comprising the steps of:

3 receiving a plurality of client requests for connections; and
4 implementing a scheduling heuristic to alleviate over-scheduling of a plurality of worker
5 threads to said plurality of client requests.

1 30. The method for enhancing performance of a multithreaded application according to Claim
2 29, wherein:

3 a first group of said worker threads are active threads, said first group being comprised of
4 changeable ones of said plurality of worker threads, and having a changeable number of said
5 changeable ones, said changeable number being at least one; and

6 said implementing a scheduling heuristic step further comprises balancing said changeable
7 number in said first group against a current workload comprised of one or more of said plurality
8 of client requests.

1 31.... The method for enhancing performance of a multithreaded application according to Claim
2 30, wherein said balancing step further comprises using an average delay.

1 32. The method for enhancing performance of a multithreaded application according to Claim
2 31, wherein said balancing step further comprises using a maximum delay.

1 33. The method for enhancing performance of a multithreaded application according to Claim
2 32, wherein said average delay and said maximum delay are configuration parameters.

1 34. The method for enhancing performance of a multithreaded application according to Claim
2 30, wherein:
3 a second group of said worker threads are blocked threads, said second group being
4 comprised of ones of said plurality of worker threads which are not in said first group; and
5 said blocked threads are stored in a Last-In, First-Out queue.

1 35. A method for enhancing performance of a multithreaded application in a computing
2 environment having a connection to a network, comprising the steps of:
3 moving connections from a pending connections queue to a first queue when each of said
4 connections are accepted;
5 moving each of said connections from said first queue to a second queue when an initial
6 data packet arrives for said connection; and

7 assigning a worker thread to each of said connections on said second queue.

1 36. A method for enhancing performance of a multithreaded application in a computing
2 environment having a connection to a network, comprising the steps of:
3 receiving input from multiple sources; and
4 merging said received input onto a single queue for scheduling.

1 37. The method for enhancing performance of a multithreaded application according to Claim
2 36, further comprising the steps of:

3 moving connections from a pending connections queue to a first queue when each of said
4 connections are accepted;
5 moving each of said connections from said first queue to said single queue when an initial
6 data packet arrives for said connection; and
7 assigning a worker thread to each of said connections on said single queue.

1 38. The method for enhancing performance of a multithreaded application according to Claim
2 37, further comprising:

3 a group of active worker threads comprised of changeable ones of a plurality of worker
4 threads, and having a changeable number of said changeable ones, said changeable number being
5 at least one; and
6 wherein said scheduling step further comprises:
7 implementing a scheduling heuristic for balancing said changeable number in said active

8 group against a current workload comprised of said client requests stored on said single queue.

1 39. A method for enhancing performance of a multithreaded application in a computing
2 environment having a connection to a network, comprising the steps of:

3 binding selected ones of a plurality of persistent connections to selected ones of a plurality
4 of worker threads, wherein an execution of said binding step results in a bound connection; and
5 unbinding selected ones of said bound connections, wherein an execution of said
6 unbinding step results in an unbound worker thread.

10 40. The method for enhancing performance of a multithreaded application according to Claim
20 39, wherein:

3 said binding step further comprises using a 2-stage queue; and
4 said unbinding step further comprises using said 2-stage queue.

1H 41. The method for enhancing performance of a multithreaded application according to Claim
2D 40, wherein:

3 said binding using said 2-stage queue step further comprises the steps of:
4 moving each of said persistent connections to said first stage when an initial data
5 packet arrives for said connection;
6 moving each of said persistent connections from said second stage to said first
7 stage when data is received for said connection; and
8 scheduling said persistent connections from said first stage; and

9 said unbinding using said 2-stage queue step further comprises the steps of:
10 moving selected ones of said bound connections from said first stage to said
11 second stage when said selected bound connection goes idle;
12 closing selected ones of said persistent connections in said second stage,
13 responsive to a maximum idle period being exceeded; and
14 making said unbound worker thread available to said subprocess for binding.

1 42. The method for enhancing performance of a multithreaded application according to Claim
2 41, wherein said unbinding step further comprises the step of:
3 closing further selected ones of said persistent connections in said second stage,
4 responsive to exceeding a maximum number of idle connections.

SEARCHED - INDEXED - SERIALIZED - FILED